

Computing Bases of Morphism Modules of Drinfeld Modules

1 Introduction

Talk about Drinfeld modules and the morphism basis

Main theorem:

Theorem 1. There is an algorithm for computing a basis of

1. $\text{Hom}(\phi, \psi)_d$ over $\mathbb{F}_q$ using $(d^\omega n^\omega \log q + dn^2 r \log q + n \log^2 q)^{1+o(1)}$ bit operations.
2. $\text{Hom}(\phi, \psi)$ over $\mathbb{F}_q[\tau^n]$ using $(n^{2\omega} r \log q + n \log^2 q)^{1+o(1)}$ bit operations.
3. $\text{Hom}(\phi, \psi)$ over $\mathbb{F}_q[x]$ using $(n^3 r^3 \log q + n \log^2 q)^{1+o(1)}$ bit operations.

2 Background

Here we introduce the necessary algorithmic background and notation. We will also introduce Drinfeld modules and morphisms between them. Throughout this work, we will fix a prime power q and a finite field $\mathbb{F}_q$ of order q . We further fix a degree n extension $\mathbb{L} \supset \mathbb{F}_q$ and an $\mathbb{F}_q$ -algebra homomorphism $\gamma : \mathbb{F}_q[x] \rightarrow \mathbb{L}$; $\ker \gamma$ is generated by a monic irreducible $\mathfrak{p} \in \mathbb{F}_q[x]$ of degree m . The situation is represented by the following tower of fields:

$$\mathbb{F}_q \xrightarrow{m} \mathbb{F}_{\mathfrak{p}} \xrightarrow{n/m} \mathbb{L}$$

2.1 Algorithmic Background

2.2 Drinfeld Modules

Definition 2.1. Let $\mathbb{L}'$ be an extension of $\mathbb{L}$. An $\mathbb{L}'$ -**morphism** of Drinfeld modules $u : \phi \rightarrow \psi$ is a $u \in \mathbb{L}'\{\tau\}$ such that $u\phi_a = \psi_a u$ for all $a \in A$.

We will typically consider $\mathbb{L}$ -morphisms, and the unqualified term *morphism* will refer to this case. The motivation for this definition can be more clearly understood in terms of the induced A -module on $\overline{\mathbb{L}}$, as the definition is equivalent to the requirement that the following diagram commutes.

$$\begin{array}{ccc} \overline{\mathbb{L}} & \xrightarrow{\phi_a} & \overline{\mathbb{L}} \\ \downarrow u & & \downarrow u \\ \overline{\mathbb{L}} & \xrightarrow{\psi_a} & \overline{\mathbb{L}} \end{array}$$

We will use $\text{Hom}(\phi, \psi)$ to denote the set of morphisms from ϕ to ψ and $\text{End}(\phi) = \text{Hom}(\phi, \phi)$. $\text{End}(\phi)$ is the centralizer for $\phi(A)$, and in particular of ϕ_x , in $\mathbb{L}\{\tau\}$. A non-zero element of $\text{Hom}(\phi, \psi)$ is referred to as an *isogeny* and two Drinfeld modules ϕ, ψ are *isogenous* if there is a non-zero member of $\text{Hom}(\phi, \psi)$. An *isomorphism* of Drinfeld modules is a $u \in \text{Hom}(\phi, \psi) \cap \mathbb{L}$. Recalling $[\mathbb{L} : \mathbb{F}_q] = n$, $\text{End}(\phi)$ contains the so-called *Frobenius endomorphism*, τ^n .

3 Previous Work on Computing Morphisms

The main works we consider here are those of Kuhn & Pink in

3.1 Wesolowski's Algorithm

In this section we give an explicit description of the algorithm given by Wesolowski in [Wes22]. Fix a basis $\{f_j\}_{j=0}^{n-1}$ for $\mathbb{L}/\mathbb{F}_q$. Set $\phi_x = \sum_{i=0}^r \Delta_i \tau^i$, $\psi_x = \sum_{i=0}^r \Lambda_i \tau^i$, and $u = \sum_{j=1}^n \sum_{k=0}^d u_{j,k} f_j \tau^i$, with $u_{j,k} \in \mathbb{F}_q$. Then $u \in \text{Hom}(\phi, \psi)_d = \{u \in \text{Hom}(\phi, \psi) \mid \deg_\tau(u) \leq d\}$ if and only if it satisfies the following relation:

$$\sum_{i=0}^r \sum_{j=0}^{n-1} \sum_{k=0}^d \Delta_i^{[k]} f_j u_{j,k} \tau^{i+k} = \sum_{i=0}^r \sum_{j=1}^n \sum_{k=0}^d \Lambda_i u_{j,k} f_j^{[i]} \tau^{i+k}. \quad (1)$$

We can compute all products $\Lambda_i f_j^{[i]}$, $\Delta_i^{[k]} f_j$ for a total bit cost of $(dn^2 r \log q)^{1+o(1)}$. Rewriting each entry in terms of the $\mathbb{F}_q$ -basis for $\mathbb{L}$

$$\begin{aligned} \Lambda_i f_j^{[i]} &= \sum_{\theta=0}^{n-1} \lambda_{i,j,\theta} f_\theta \\ \Delta_i^{[k]} f_j &= \sum_{\theta=0}^{n-1} \delta_{i,j,k,\theta} f_\theta \end{aligned}$$

allows us to define a linear system over $\mathbb{F}_q$ coming from

$$\sum_{i=0}^r \sum_{j=0}^{n-1} \sum_{k=0}^d \sum_{\theta=0}^{n-1} \delta_{i,j,k,\theta} u_{j,k} f_\theta \tau^{i+k} = \sum_{i=0}^r \sum_{j=1}^n \sum_{k=0}^d \sum_{\theta=0}^{n-1} \lambda_{i,j,\theta} u_{j,k} f_\theta \tau^{i+k}. \quad (2)$$

We therefore obtain a linear equation in the $\mathbb{F}_q$ variables $u_{j,k}$ from extracting the coefficients of each $f_j \tau^i$ for $0 \leq j < n$ and $0 \leq i \leq d+r$, allowing us to construct a $(d+r)n \times (d+1)n$ linear system over $\mathbb{F}_q$ whose kernel is exactly $\text{Hom}(\phi, \psi)_d$. Constructing the system from equation (2) costs $O(dn^2 r)$ operations in $\mathbb{F}_q$ as well as $O(r+n)$ applications of the Frobenius. Pre-computing the Frobenius takes the usual $(n \log^2 q)^{1+o(1)}$ bit cost; with the overall bit cost to construct and solve the system being $(d^\omega n^\omega \log q + dn^2 r \log q)^{1+o(1)}$, we obtain the complexity stated in item 1 of theorem 1.

4 An Explicit Description of the Kuhn & Pink Algorithm

One can repeat a similar procedure for bases of $\text{Hom}(\phi, \psi)$ over $\mathbb{F}_q[\tau^n]$ and $\mathbb{F}_q[x]$, and indeed such a computation was discussed in [KP22]. In the former case, $\{f_j \tau^k\}_{1 \leq j \leq n}^{0 \leq k < n}$ is an $\mathbb{F}_q$ -basis for $\mathbb{L}\{\tau\}$

with coefficients in $\mathbb{F}_q[\tau^n]$; so we can write $u = \sum_{j=1}^n \sum_{k=0}^{n-1} \alpha_{j,k} f_j \tau^k$ with $\alpha_{i,j} \in \mathbb{F}_q[\tau^n]$. Then $\text{Hom}(\phi, \psi)$

consists of such u satisfying the relation

$$\sum_{i=0}^r \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} \Delta_i^{[k]} f_j \alpha_{j,k} \tau^{i+k} = \sum_{i=0}^r \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} \Lambda_i \alpha_{j,k} f_j^{[i]} \tau^{i+k} \quad (3)$$

As seen previously, we can compute a basis decomposition of all $\Lambda_i f_j^{[i]}$ and $\Delta_i^{[k]} f_j$ at a bit cost of $(rn^3 \log q)^{1+o(1)}$. Then equation 4 becomes:

$$\sum_{i=0}^r \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} \sum_{\theta=0}^{n-1} \delta_{i,j,k,\theta} \alpha_{j,k} \tau^{i+k} = \sum_{i=0}^r \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} \sum_{\theta=0}^{n-1} \lambda_{i,j,\theta} \alpha_{j,k} \tau^{i+k} \quad (4)$$

By rewriting the above expression in terms of the basis elements $f_j \tau^k$ for $j < n$ and $k < n$ with coefficients $\rho_{i,j,k,\ell} \in \mathbb{F}_q[\tau^n]$, we obtain:

$$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} \sum_{\ell=0}^{n-1} \rho_{i,j,k,\ell} \alpha_{k,\ell} f_j \tau^i = 0$$

Then the coefficients $\alpha_{k,\ell}$ of u can be solved for by constructing an $n^2 \times n^2$ linear system over $\mathbb{F}_q[\tau^n]$, whose coefficients have degree at most $\lfloor \frac{r}{n} \rfloor + 1$, of the form

$$\begin{bmatrix} \rho_{0,0,0,0} & \cdots & \rho_{0,0,n-1,n-1} \\ \vdots & \ddots & \vdots \\ \rho_{n-1,n-1,0,0} & \cdots & \rho_{n-1,n-1,n-1,n-1} \end{bmatrix} \begin{bmatrix} \alpha_{0,0} \\ \vdots \\ \alpha_{n-1,n-1} \end{bmatrix} = 0$$

We may then take advantage of pre-existing algorithms for computing a minimal nullspace basis for the system with a complexity of $O(n^{2\omega-1}r)$ field operations in $\mathbb{F}_q$ [ZLS12], adding a cost of $(n^{2\omega}r \log q)$ in our bit complexity model. Constructing the system costs at most $(n^3r \log q)^{1+o(1)}$ bit operations, and adding in the usual pre-computation for Frobenius maps gives the stated complexity in item 2 of theorem 1.

5 Computing Morphisms over $\mathbb{F}_q[x]$

We can repeat the core idea of this algorithm once again for a basis over $\mathbb{F}_q[x]$, though some more care must be taken to address technical details. More explicitly, we view $\mathbb{L}\{\tau\}$ as an $\mathbb{F}_q[x]$ -module with the action $x * u = u\phi_x$, and note that u is not required to lie inside $\text{Hom}(\phi, \psi)$. Write $u = \sum_{i=0}^{r-1} \sum_{j=0}^{n-1} u_{i,j} f_j \tau^i$ with $u_{i,j} \in \mathbb{F}_q[x]$. Then if $u \in \text{Hom}(\phi, \psi)$ we must have

$$u\phi_x = x * u = \sum_{i=0}^{r-1} \sum_{j=0}^{n-1} x u_{i,j} f_j \tau^i = \psi_x u = \sum_{i=0}^{r-1} \sum_{j=0}^{n-1} \sum_{k=0}^r \Lambda_k f_j^{[k]} u_{i,j} \tau^{i+k} \quad (5)$$

Constructing the linear system as we did previously can be done by computing coefficients $\rho_{t,i,j} \in \mathbb{F}_q[x]$ such that

$$\tau^t = \sum_{w=0}^{r-1} \sum_{z=0}^{n-1} \rho_{t,w,z} f_z \tau^w$$

for $r \leq t < 2r$. Recalling the definition of the $\mathbb{F}_q[x]$ -action on $\mathbb{L}\{\tau\}$ $x\tau^i = \tau^i\phi_x$, we can re-arrange the expression to obtain the following recurrence, for any $i \geq 0$

$$\tau^{r+i} = \sum_{k=0}^{r-1} \left(-\frac{\Delta_k}{\Delta_r} \right)^{[i]} \tau^{k+i} + x\tau^i. \quad (6)$$

This can be evaluated at most $r-1$ times to allow the extraction of the required coefficients $\rho_{t,i,j}$. Note that $\deg \rho_{t,i,j} \leq 1$. Each such evaluation requires $O(r)$ applications of the Frobenius, and $O(r^2)$ operations in $\mathbb{L}$. This gives an overall bit complexity of $(r^3n \log q)^{1+o(1)}$ for computing all required $\rho_{t,i,j}$. Then equation (5) yields:

$$\sum_{i=0}^{r-1} \sum_{j=0}^{n-1} x u_{i,j} f_j \tau^i = \sum_{i=0}^{r-1} \sum_{j=0}^{n-1} \sum_{k=0}^r \sum_{w=0}^{r-1} \sum_{z=0}^{n-1} \Lambda_k f_j^{[k]} u_{i,j} \rho_{i+k,w,z} f_z \tau^w \quad (7)$$

In this case, it suffices to compute the decomposition of $\Lambda_k f_j^{[k]} = \sum_{\theta=0}^{n-1} \lambda_{k,j,\theta} f_\theta$ for $k \leq r$, $j \leq n$ for a total bit cost of $(n^2 r \log q)^{1+o(1)}$. Using the above equation, we can determine the coefficients of the basis elements $f_z \tau^w$ in terms of the indeterminates u_{ij} , which allows the extraction of an $nr \times nr$ linear system, with entries in $\mathbb{F}_q[x]_1$, in the usual manner. This system can then be solved over $\mathbb{F}_q[x]$ with a bit complexity of $(n^\omega r^\omega \log q)^{1+o(1)}$. Generating the system from (7) costs $(r^3 n^3 \log q)^{1+o(1)}$, which gives an overall cost of $(r^3 n^3 \log q + n \log^2 q)^{1+o(1)}$. This completes the analysis of the algorithms of theorem 1.

References

- [KU08] Kiran Kedlaya and Christopher Umans. “Fast Polynomial Factorization and Modular Composition”. In: *SIAM Journal on Computing* 40 (Jan. 2008). DOI: 10.1137/08073408X.
- [ZLS12] W. Zhou, G. Labahn, and A. Storjohann. “Computing Minimal Nullspace Bases”. In: *Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation*. ISSAC ’12. Grenoble, France: Association for Computing Machinery, 2012, pp. 366–373. ISBN: 9781450312691. DOI: 10.1145/2442829.2442881. URL: <https://doi.org/10.1145/2442829.2442881>.
- [CB17] X. Caruso and J. Le Borgne. “Fast multiplication for skew polynomials”. In: *ISSAC’17*. ACM, 2017, pp. 77–84.
- [PW17] S. Puchinger and A. Wachter-Zeh. “Fast operations on linearized polynomials and their applications in coding theory”. In: *J. Symb. Comput.* (2017).
- [KP22] N. Kuhn and R. Pink. “Finding endomorphisms of Drinfeld modules”. In: *Journal of Number Theory* 232 (2022). Special Issue: David Goss Memorial Issue, pp. 118–154. ISSN: 0022-314X. DOI: <https://doi.org/10.1016/j.jnt.2021.02.013>. URL: <https://www.sciencedirect.com/science/article/pii/S0022314X21001852>.
- [Wes22] B. Wesolowski. *Computing isogenies between finite Drinfeld modules*. Cryptology ePrint Archive, Paper 2022/438. <https://eprint.iacr.org/2022/438>. 2022. URL: <https://eprint.iacr.org/2022/438>.